MAIN FRAME PLUS★WARE™ NEWS
A PUBLICATION OF STSC, INC.
December 1986 Issue 6

Compiler Cuts CPU Charges for A.C. Nielsen

The APL★PLUS® Compiler, which is installed at MVS and VM sites around the world, is proving again and again that it can be a cost-effective means of speeding up real-life APL applications. This article summarizes the improvements obtained by compiling two applications written by clients of STSC.

Market Research Analysis

At the A. C. Nielsen Company of Canada, located in Toronto, the APL★PLUS System is used to run the ERB (Electronic Report Book) workspace, an interactive marketing research analysis application. It reads data from large VSAM data sets and produces reports and plots. This application is used very heavily; nearly 100 people use it for about 20% of their working day. Before compilation, the application consumed about 12 CPU hours per month, costing nearly $10,000/month at the internal billing rate. It was hoped that, by speeding up this and other APL applications, the compiler would save enough CPU time to pay for itself.

When the application was first run under the APL★PLUS System, the monitoring facility (□*MF*) proved valuable, revealing that a comment-stripping program consumed nearly a third of the time in a test run. The application was modified so the comment stripping would be done before saving the workspace, and this faster version was considered a base for further improvements.

Originally, it was feared that most of the time in this application was spent in VSAM I/O, but the monitoring facility revealed numerous opportunities for the compiler and FASTFNS to speed up execution. Out of 96 functions that were called, 12 functions were compiled and 3 were modified to use FASTFNS. The following table gives the speedup factor for each function, arranged from most to least time consuming:

Function	Speedup Factor	Function	Speedup Factor
GET	1.4	*ECO*	2.2
KEYDECODE	2.4	*PRINT*1	1.7
BINARY	2.7	*DELLEFT*	32.0
PRINTER	1.1	*CH*	6.9
SPLITCCBRD	3.5	*PRINTCMS*1	1.8
OUT	1.9	*CCMODDTYPE*	1.5
OUTCMS	2.9	*RESHAPE*	2.4
GETM	2.1		

In identical test runs on the APL★PLUS System, the compiled version of the application is nearly twice as fast as the interpreted version. Using the improved version of the ERB application saves nearly $5,000 per month in CPU charges.

The ERB application originally took about 2 man-years to develop. It was sped up using the compiler in about a month, with half of that time spent learning about the compiler. Tom Saint, who wrote the workspace, says he feels that "the compiler is a good product, nicely packaged, that does what it claims to do." He believes that the compiler has removed one of the obstacles to using APL for applications, and that if faster access to operating system files were available, more applications would be developed in APL.

Insurance Calculations

We recently performed a benchmark of the compiler for an insurance company located in Montreal. Their DEFMPST application reads CMS files, calculates reserves for group deferred income policies, and writes the results to other CMS files. Although it is used only 7 to 20 times a year, each run consumes an enormous amount of CPU time—4,730 CPU seconds on STSC's IBM 3033U computer. We compiled 5 of the most CPU intensive functions out of 17 that were executed and obtained the following speedup factors:

Function	Speedup Factor
RETIRED	3.5
DEATHBEN	6.8
ANN	4.7
FACTORS	2.6
SPECIALDB	1.8

Overall, the compiled version of the application is 3.6 times faster than the interpreted version, amounting to a savings of about 3,400 CPU seconds per run.

In this benchmark, we also recorded how much time it took to compile functions (these were rather large functions, totaling 3,400 tokens). In this case, the cost of compilation is more than paid back by the savings from running the application just once. In the course of a year, the compiled version of this application might save as much as 20 CPU hours.

Your Application?

STSC's customers use the compiler for a variety of reasons: to reduce CPU charges, to save time spent waiting for APL applications to finish, and to postpone purchase of a larger computer . If you want your APL applications to run faster, let STSC's state-of-the-art APL technology help you stretch your computing capacity and reduce costs. Contact your STSC representative today to find out more about the APL★PLUS Compiler!

Useful Utility Functions

by Jim Weigang

Some user-defined APL functions are so useful that they find their way into practically every workspace. Such programs provide general-purpose capabilities missing from APL's set of primitive operations, and it is reasonable to think of them as new "primitive" operations. This comparison is especially apt if you use precompiled functions from STSC's FASTFNS workspace, because they run almost as fast as functions built into the interpreter. This article discusses several useful utility functions from the FASTFNS workspace.

Novice APL users are often disappointed when they discover that catenation joins only conforming arrays. A common requirement is to join two arrays top-to-bottom, padding with blanks to produce a rectangular matrix. The *OVER* function does just that:

```
      'A' OVER 'USEFUL' OVER 'FUNCTION'
A
USEFUL
FUNCTION
```

The arguments can be scalars, vectors, or matrices of character or numeric type; the result is a matrix.

Many programs accept a list of names as an argument. It is customary to allow such a list to be represented as either a matrix with a name on each row, or as a vector with blanks separating each name. Within the program, the *MATRIFY* function can be used to convert the argument to a matrix:

```
      MATRIFY '  A   USEFUL FUNCTION '
A
USEFUL
FUNCTION
```

If the argument is already a matrix, *MATRIFY* returns it unchanged. *MATRIFY* has an optional left argument that specifies what characters are to be considered delimiters. For example:

```
      ',;/' MATRIFY 'ONE TWO,THREE/FOUR'
ONE TWO
THREE
FOUR
```

Searching is another common operation, and APL provides a primitive function, dyadic iota ($A \iota B$), to search for scalars. This does not solve all searching problems, however; for example, it is often necessary to search for a particular row in a matrix. The *MATIOTA* function is similar to dyadic iota, but applies to the rows of character matrices. For example, if variable N is a matrix of names, and T is a matrix of telephone numbers, *MATIOTA* and indexing can be used to find the number for a particular person:

```
      T[N MATIOTA'JONES';]
(413) 268-7500
```

MATIOTA can also be used in place of dyadic iota to adapt several common idioms to work with character matrices. For example, duplicate names can be removed from a matrix M with `((⍳1↑⍴M)=M MATIOTA M)⌿M`, and names A can be removed from list B with $(0=A$ *MATIOTA* $B)/B$.

Like the *MATRIFY* function, the *VTOM* function converts a vector to a matrix, but *VTOM* uses the first character of its argument as the delimiter and produces blank rows in the result if there are multiple consecutive delimiters. The *MTOV* function is the inverse of *VTOM*; it converts a matrix to a delimited vector. Together, *VTOM* and *MTOV* can be used in a "dual" to perform matrix operations on delimited vectors. For example, to sort a list of words stored in a blank-delimited vector, you could use:

```
      M←VTOM ' ONE THREE FOUR'
      M←M[⎕AV⍋M;]
      ' 'MTOV M
FOUR ONE THREE
```

[In mathematics, a "dual" is when something is done by applying some function (call it F), doing something to the result, and then applying the inverse of F (call it G):

```
G OP F X
```

This is a general purpose technique that comes up again and again. If you don't know how to perform an operation on one data structure, but you know how to perform the operation on another structure and can map between the two structures, a dual can be used to get the job done. Can you think of any duals you have used in programs? Can you think of any *double* duals?]

Character matrices are usually stored with each row left justified, but for some kinds of operations it's convenient to have the rows right justified. The *RJUST* function can be used to do this. For instance, a list of variables whose names end in "ΔDOC" can be produced with:

```
      R←1↑⍴N←⎕NL 2
      (((R,¯4)↑RJUST N)∧.='ΔDOC')⌿N
RJUSTΔDOC
LJUSTΔDOC
```

RJUST has an inverse function, *LJUST*, that left justifies a matrix. Together, they can be used in a dual to append a new character to the end of each row of a matrix:

```
      LJUST(RJUST M),'.'
FOUR.
ONE.
THREE.
```

Another common operation is deleting excess blanks, performed by the *DEB* function. One thing that frequently needs deblanking is the result of the monadic format function ($\blacktriangledown A$). When a matrix is formatted, the same width is used for each column, which usually results in superfluous blanks that can be removed by *DEB*. For example:

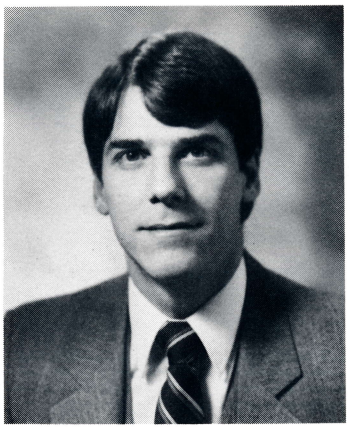
```
      ▼M←2 4↑10000
10000    0    0    0
    0    0    0    0
      DEB ▼M
10000 0 0 0
    0 0 0 0
```

The more compact form is especially helpful when it permits each row to be displayed on a single line.

OVER, *MATRIFY*, *MATIOTA*, *VTOM*, *MTOV*, *RJUST*, *LJUST*, and *DEB*, together with many other useful utility functions, can be found in the FASTFNS workspace in library 3. Documentation for each of these functions is available in the workspace, stored in variables with names

continued on page 4

Meet the Team



Jerry Turner

Jerry Turner is the National Sales Manager for STSC's mainframe and large-scale UNIX™ APL★PLUS System products. His team of account representatives are STSC's main line of contact for corporate customers throughout the United States. Jerry also plays a major role in developing and maintaining relationships with STSC's international mainframe dealers and clients.

"Building strong relationships with our customers is what it's all about," according to Jerry. "Our message is that we want to help our clients with our entire range of services—micro and mainframe software, training, consulting—whatever they need. What we have to offer are the highest quality APL products and services available anywhere. Our leadership in the APL market makes us an attractive partner for companies that are serious about APL."

Jerry is looking forward to the new organization he has planned for 1987, which will center on assigning account managers to specific industries. "A representative working in one specific field—say, investment banking or actuarial analysis—will learn how APL is used in that specialty. This industry expertise, going far beyond general APL expertise, adds even more value to the relationship with the client."

Jerry has been with STSC for over six years. He joined the company's publications department in 1980, counting among his accomplishments there the popular tutorial *APL is Easy!*, which is now included as part of every APL★PLUS System for the PC and Macintosh and is soon to be published by John Wiley and Sons. He moved to customer technical support in STSC's Washington, D.C., branch office in 1982, later becoming an account manager. In 1984 and 1985, Jerry was responsible for marketing STSC's mainframe products in the international marketplace. He became National Sales Manager in 1986.

Jerry holds B.A. and M.A. degrees in German Studies as well as an MBA, with emphasis in marketing, from the University of Maryland. He has lived abroad extensively, completing coursework at the Ludwig Maximilian Universitaet in Munich and the Johannes Gutenberg Universitaet in Mainz, West Germany. He has presented papers at a number of technical conferences, including the Share European Association (SEAS), Guide Latin America, and the International Marketing Conference of the American Marketing Association (AMA).

Jerry lives in suburban Washington, D.C., with his wife Jo and two miniature schnauzers.

Product Announcements

SHAREFILE/AP for MVS

STSC's popular SHAREFILE™ System is now available as an auxiliary processor for APL2 running under MVS. SHAREFILE/AP for MVS is a secure, shared access, component-based APL file system that uses standard IBM interfaces to provide APL data manipulation and storage features without modifications to APL2.

APL★PLUS Compiler, Release 1.1

A new version of the compiler was shipped to all current compiler sites in July and August of 1986. The new version includes minor improvements and corrections to some behavior, and five new FASTFNS, pre-compiled APL functions that are at least twice as fast as their standard APL alternatives.

□FMT for APL2

Now STSC's popular business report formatting facility, □*FMT*, is available as an add-on to APL2. □*FMT* has been implemented as an APL2 External Function executed via Associated Processor 11. The basic behavior of □*FMT* under APL2 is identical to its counterpart on the APL★PLUS System running under IBM's VS APL product. Currently, □*FMT* is available for APL2 running under both VM and MVS.

For more information about these new STSC products, call Jerry Turner, National Sales Manager, at (301) 984-5112.

APL★PLUS SQL Interface

In the VM environment, STSC provides a new interface to IBM's SQL/DS Relational Database Management System. The new Interface product is modeled after the APL2/SQL auxiliary processor and performs any SQL/DS operation allowed under APL2. It is fully compatible with the APL2/SQL interface and includes a workspace with cover functions for easy interaction with SQL/DS.

MAIN FRAME PLUS★WARE™ NEWS
A PUBLICATION OF STSC, INC.

Mainframe PLUS★WARE News is published by STSC, Inc. and is distributed free of charge to the company's customers. Comments regarding the newsletter should be sent to:

Christopher Bigler, Editor
STSC, Inc.
2115 East Jefferson St.
Rockville, MD 20852

©1986 STSC, Inc. All rights reserved. Printed in U.S.A.

APL★PLUS is a registered trademark of STSC, Inc.
PLUS★WARE and SHAREFILE are trademarks of STSC, Inc.
UNIX is a trademark of AT&T Bell Laboratories.

Improved, Low-Cost APL Training for 1987

STSC is beginning the new year with a full series of three-day classes and one-day special topic seminars on the APL language and the use of STSC products. Classes will be held in Rockville, Maryland, at STSC's corporate headquarters, and at modern seminar facilities in New York City.

Classes have been streamlined and are now three days instead of four. Starting in 1987, each regular three-day class will be followed by an optional one-day follow-up class, which will allow students the opportunity to review what they have learned and are putting to practical use on application design or implementation techniques.

The cost of classes had been reduced to $675 per person for three-day classes. One-day, special topic seminars are $225. STSC will also offer special two-day and five-day classes on various topics at your request when arranged in advance with a minimal guaranteed number of attendees. Special classes can also be arranged at your company's site.

Following is the schedule for the first quarter of 1987. Additional dates will be provided in upcoming issues of this newsletter.

Introductory APL with file handling and report formatting techniques:

January 12-16	Rockville
January 26-30	New York City

Pocket APL one-day tutorial:

January 20	Rockville
January 23	New York City
February 17	Rockville
February 19	New York City
March 10	Rockville
March 12	New York City

APL★PLUS System on the Apple Macintosh with desktop publishing techniques:

February 9-13	Rockville
March 16-20	New York City

Intermediate APL Programming incorporating advanced features and techniques for programming with APL★PLUS for the PC:

February 23-27	New York City
March 2-6	Rockville

Later in the year, a four-day course on using the APL★PLUS System in the UNIX environment and a course on documentation design and technical writing skills will be offered. Announcements of upcoming classes will be published in this newsletter. For more information about topics, schedules, locations, and class fees, contact Tama Traberman, Educational Services Manager, in New York City at 212-984-0730, or Bonni Hardy at 800-638-6660.

STSC Announces APL★PLUS System for the Macintosh

The APL★PLUS System for the Macintosh is a full-featured APL language interpreter, tailored to the unique Macintosh computer environment. The system is compatible with STSC's APL★PLUS System for IBM PCs and compatibles, allowing users with existing applications to convert them to run on the Macintosh.

The APL★PLUS System takes advantage of standard Macintosh features such as the Hierarchical File System (HFS), QuickDraw graphics, dialog boxes, menus, cut-and-paste full-screen editing, as well as Clipboard and Scrapbook data exchange features. Common desk accessories such as Paint Grabber, features such as horizontal and vertical scrolling, and applications such as Switcher can also be used from the APL environment.

The APL★PLUS System is capable of handling very large applications because the APL workspace and APL objects are limited in size only by available memory, which can expand to four megabytes. STSC provides its popular enhancements to the APL language, along with component and native file systems, error handling, and extensive data formatting facilities. A library of tutorial and utility software is also provided along with routines for configuring the APL environment, accessing the communications port, and preparing custom menus. Since the APL★PLUS System for the Macintosh is compatible with STSC's APL★PLUS Systems across hardware architectures, built-in terminal emulation, file transfer facilities, and upload and download capabilities make it convenient to move data and programs to and from other APL★PLUS Systems on minis, micros, and mainframes.

The APL★PLUS System for the Macintosh includes a complete set of software and documentation, an extensive collection of utility and demonstration workspaces, and special APL fonts, which allow the Macintosh to display and print the complete APL character set on both the screen and Imagewriter or LaserWriter printers.

STSC is now accepting orders for the new product, which will be shipped in December 1986. The U.S. retail price for the entire APL★PLUS System for the Macintosh is $395. Current users of PortaAPL can upgrade to the APL★PLUS System for $80 until January 31, 1987.

The APL★PLUS System for the Macintosh operates on a Macintosh or Macintosh Plus, with at least 512K RAM and one disk drive. The system is available through local dealers or by contacting STSC at (800) 592-0050, or at (301) 984-5123 in Maryland or Canada.

Useful Utility Functions *continued from page 2*

ending '*ΔDOC*'. If you have suggestions for commonly-used functions that aren't in the FASTFNS workspace, send them to the FASTFNS Steward, care of STSC.

Thought of any double duals yet? Here's one in the APL version of the *FROM* FASTFN:

```
FROM:(,⍵)[1+⍉(⍴⍵)⊥⍉∝-1]
```

One dual involves transpose, which is its own inverse function; the other involves subtracting and adding 1.